WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Wednesday, November 24, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ	
☐	L9	L5 and "power of 2"	0
☐	L8	L5 and L2	19
☐	L7	L6 and L2	0
		DB=EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L6	L5	9
		DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L5	L4 and (second near2 (counter or address indicator))	462
☐	L4	L3 and (packet or instruction unit or execution unit)	5275
☐	L3	(counter or address indicator) same instruction	30368
		DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ	
☐	L2	L1 or (712/208 |712/209 |712/210 |712/211 |712/212 |712/213 |712/230 |712/231 |712/232).ccls.	2454
☐	L1	(717/127 |717/128 |717/129 |717/130 |717/131).ccls.	1019

END OF SEARCH HISTORY

Terms used **instruction counter**

Found **178** of **147,060**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

☐ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 178 Result page: **1** 2 3 4 5 6 7 8 9 next

Relevance scale ■■■■■

1 A software instruction counter

J. M. Mellor-Crummey, T. J. LeBlanc

April 1989 **ACM SIGARCH Computer Architecture News , Proceedings of the third international conference on Architectural support for programming languages and operating systems**, Volume 17 Issue 2

Full text available: pdf(997.70 KB) Additional Information: full citation, abstract, references, citings, index terms

Although several recent papers have proposed architectural support for program debugging and profiling, most processors do not yet provide even basic facilities, such as an instruction counter. As a result, system developers have been forced to invent software solutions. This paper describes our implementation of a software instruction counter for program debugging. We show that an instruction counter can be reasonably implemented in software, often with less than 10% execution overhead. Ou ...

2 Hardware-assisted replay of multiprocessor programs

David F. Bacon, Seth Copen Goldstein

December 1991 **ACM SIGPLAN Notices , Proceedings of the 1991 ACM/ONR workshop on Parallel and distributed debugging**, Volume 26 Issue 12

Full text available: pdf(1.20 MB) Additional Information: full citation, references, citings, index terms

3 Replay for concurrent non-deterministic shared-memory applications

Mark Russinovich, Bryce Cogswell

May 1996 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1996 conference on Programming language design and implementation**, Volume 31 Issue 5

Full text available: pdf(968.81 KB) Additional Information: full citation, abstract, references, citings, index terms

Replay of shared-memory program execution is desirable in many domains including cyclic debugging, fault tolerance and performance monitoring. Past approaches to repeatable execution have focused on the problem of re-executing the shared-memory access patterns in parallel programs. With the proliferation of operating system supported threads and shared memory for uniprocessor programs, there is a clear need for efficient replay of concurrent applications. The solutions for parallel systems can b ...

Keywords: instruction counter, non-determinism, repeatable execution, shared memory

4 The execute operations—a fourth mode of instruction sequencing
F. P. Brooks
March 1960 **Communications of the ACM**, Volume 3 Issue 3
Full text available: pdf(415.00 KB) Additional Information: full citation, references

5 A survey of rollback-recovery protocols in message-passing systems
E. N. (Mootaz) Elnozahy, Lorenzo Alvisi, Yi-Min Wang, David B. Johnson
September 2002 **ACM Computing Surveys (CSUR)**, Volume 34 Issue 3
Full text available: pdf(549.68 KB) Additional Information: full citation, abstract, references, citings, index terms, review

This survey covers rollback-recovery techniques that do not require special language constructs. In the first part of the survey we classify rollback-recovery protocols into *checkpoint-based* and *log-based. Checkpoint-based* protocols rely solely on checkpointing for system state restoration. Checkpointing can be coordinated, uncoordinated, or communication-induced. *Log-based* protocols combine checkpointing with logging of nondeterministic events, encoded in tuples call ...

Keywords: message logging, rollback-recovery

6 A microprogrammed implementation of EULER on IBM system/360 model 30
Helmut Weber
September 1967 **Communications of the ACM**, Volume 10 Issue 9
Full text available: pdf(1.34 MB) Additional Information: full citation, references, citings, index terms

7 Parallel RAMs with owned global memory and deterministic context-free language recognition
Patrick W. Dymond, Walter L. Ruzzo
January 2000 **Journal of the ACM (JACM)**, Volume 47 Issue 1
Full text available: pdf(223.64 KB) Additional Information: full citation, abstract, references, index terms, review

We identify and study a natural and frequently occurring subclass of Concurrent Read, Exclusive Write Parallel Random Access Machines (CREW-PRAMs). Called Concurrent Read, Owner Write, or CROW-PRAMS, these are machines in which each global memory location is assigned a unique "owner" processor, which is the only processor allowed to write into it. Considering the difficulties that would be involved in physically realizinga full CREW-PRAM model and demonstrate i ...

Keywords: CROW-PRAM, DCFL recognition, owner write, parallel algorithms

8 Design problems in emulating the MIX computer on the Microdata 1600
T. Don Dennis, O. G. Johnson
September 1976 **Proceedings of the 9th annual workshop on Microprogramming**
Full text available: pdf(388.01 KB) Additional Information: full citation, abstract, references, index terms

This paper presents an overview of an emulator for the MIX computer written in Microdata 1600 microcode. The MIX computer thus emulated is a variant of the original MIX computer as described in Volume 1 of The Art of Computer Programming by Donald Knuth. Basic

changes involve the utilization of 8 bit bytes along with the ASCII character code.

9 System Specifications for the DYSEAC
Alan L. Leiner
April 1954 **Journal of the ACM (JACM)**, Volume 1 Issue 2
Full text available: pdf(1.40 MB) Additional Information: full citation, citings, index terms

10 SPAM: a microcode based tool for tracing operating system events
Stephen W. Melvin, Yale N. Patt
December 1987 **Proceedings of the 20th annual workshop on Microprogramming**
Full text available: pdf(405.55 KB) Additional Information: full citation, abstract, references, citings, index terms

We have developed a tool called SPAM (for System Performance Analysis using Microcode), based on microcode modifications to a VAX 8600, that traces operating system events as a side-effect to normal execution. This trace of interrupts, exceptions, system calls and context switches can then be processed to analyze operating system behavior for the purpose of debugging, tuning or development. SPAM allows measurements to be made on a fully operating UNIX system with little perturbation (typica ...

11 Efficiently counting program events with support for on-line queries
Thomas Ball
September 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 16 Issue 5
Full text available: pdf(784.76 KB) Additional Information: full citation, abstract, references, citings, index terms

The ability to count events in a program's execution is required by many program analysis applications. We represent an instrumentation method for efficiently counting events in a program's execution, with support for on-line queries of the event count. Event counting differs from basic block profiling in that an aggregate count of events is kept rather than a set of counters. Due to this difference, solutions to basic block profiling are not well suited to event counting. Our algorithm fin ...

Keywords: control-flow graph, counting, instrumentation

12 Evaluation and performance of computers: the program monitor—a device for program performance measurement
C. T. Apple
August 1965 **Proceedings of the 1965 20th national conference**
Full text available: pdf(856.22 KB) Additional Information: full citation, abstract, citings, index terms

IN 1961 A GROUP was established within IBM to test systems programs before they were released for customer usage. The goal of this group was to assure IBM management that each program released would be satisfactorily usable by the customer. One step taken by this group was to develop a monitor device which would permit programmers to record information being handled by the CPU during execution. Their intent was to use this recorded information to analyze the basic nature of progr ...

13 Microprogramming revisited
Michael J. Flynn, M. Donald McLaren
January 1967 **Proceedings of the 1967 22nd national conference**
Additional Information:

Full text available: pdf(770.01 KB) full citation, abstract, references, citings, index terms

It is the objective of this paper to briefly trace the history of the idea and the difficulties involved with defining or implementing it. In doing this, we first consider the general control problem and instruction formats. Next, storage implementations of the control function are considered and a restricted definition of microprogramming is proposed. This is then evaluated from a technological, architectural and programming point of view. We hope to show that our (demanding) definition of ...

14 Introducing computer concepts by simulating a simple computer
Robert A. Campbell
September 1996 **ACM SIGCSE Bulletin**, Volume 28 Issue 3

Full text available: pdf(217.65 KB) Additional Information: full citation, abstract, citings, index terms

The simulated computer consists of (1) main memory, (2) a register known as the accumulator, (3) a central processing unit (CPU), and (4) an instruction counter. This computer recognizes 8 op codes (Halt, Load, Store, Add, Subtract, Read, Write, and Branch On Zero). The computer is simulated by creating a program in Pascal or C++. This program simulates the execution of programs written by students, such as adding two numbers and printing their sum. Student programs are written in machine langua ...

15 Optimal tracing and incremental reexecution for debugging long-running programs
Robert H. B. Netzer, Mark H. Weaver
June 1994 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1994 conference on Programming language design and implementation**, Volume 29 Issue 6

Full text available: pdf(1.34 MB) Additional Information: full citation, references, citings, index terms

16 Race Frontier: reproducing data races in parallel-program debugging
Jong-Deok Choi, Sang Lyul Min
April 1991 **ACM SIGPLAN Notices , Proceedings of the third ACM SIGPLAN symposium on Principles and practice of parallel programming**, Volume 26 Issue 7

Full text available: pdf(1.05 MB) Additional Information: full citation, references, citings, index terms

17 Special session on memory wall: Fighting the memory wall with assisted execution
Michel Dubois
April 2004 **Proceedings of the first conference on computing frontiers on Computing frontiers**

Full text available: pdf(231.18 KB) Additional Information: full citation, abstract, references, index terms

Assisted execution is a form of simultaneous multithreading in which a set of auxiliary "assistant" threads, called *nanothreads*, is attached to each thread of an application. Nanothreads are lightweight threads which run on the same processor as the main (application) thread and help execute the main thread as fast as possible. Nanothreads exploit resources that are idled in the processor because of hazards due to program dependencies and memory access delays.Assisted execution has the po ...

Keywords: cache memories, latency tolerance, prefetching, simultaneous multithreading, superscalar processors

18 Debugging standard ML without reverse engineering
Andrew P. Tolmach, Andrew W. Appel

May 1990 **Proceedings of the 1990 ACM conference on LISP and functional programming**

Full text available: pdf(1.29 MB) Additional Information: full citation, abstract, references, citings, index terms

We have built a novel and efficient replay debugger for our Standard ML compiler. Debugging facilities are provided by instrumenting the user's source code; this approach, made feasible by ML's safety property, is machine-independent and back-end independent. Replay is practical because ML is normally used functionally, and our compiler uses continuation-passing style; thus most of the program's state can be checkpointed quickly and compactly using call-with-current-continuation. Together, ...

19 Record/replay for nondeterministic program executions

Michiel Ronsse, Koen De Bosschere, Mark Christiaens, Jacques Chassin de Kergommeaux, Dieter Kranzlmüller

September 2003 **Communications of the ACM**, Volume 46 Issue 9

Full text available: pdf(106.09 KB) Additional Information: full citation, abstract, references, citings, index terms

Controlling the nondeterministic features within multithreaded and highly responsive applications enables the continued use of all traditional software development techniques.

20 Formal papers: A microprogrammed implementation of a block structured architecture

Michael J. Lutz, Michael J. Manthey

September 1972 **Conference record of the 5th annual workshop on Microprogramming**

Full text available: pdf(919.89 KB) Additional Information: full citation, abstract, references, citings

This paper reports our current progress at SUNY/Buffalo in building an emulator for a block structured architecture. We briefly describe the overall research effort and the architecture of the target machine. We next discuss the mapping of the emulator program onto the available resources and the logical structure of the emulator. Finally we describe the current and projected micro-level instrumentation of the emulator which forms the basis for future experimentation with the architecture.

CiteSeer Find: second counter and Documents Citations

Searching for **second counter and packet instruction**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
500 documents found. **Order: relevance to query.**

The Undecidability Of Second Order Linear Logic Without.. - Lafont (1995) (Correct) (8 citations)
The Undecidability Of **Second** Order Linear Logic Without Exponentials Yves
is also undecidable, using an encoding of two-**counter** machines originally due to Kanovich. The
lmd.univ-mrs.fr/pub/lafont/mall2.ps.Z

Effective Compiler Support for Predicated Execution .. - Mahlke, Lin, Chen, .. (1992) (Correct) (132 citations)
size with conventional if-conversion techniques. **Second**, speculative execution is difficult to combine
by comparing the contents of a loop iteration **counter** to the loop bound. Figures 1a and 1b show a
VLIW compilers must expose increasing amounts of **instruction** level parallelism (ILP)Typically, global
cardit.et.tudelft.nl/~steven/ilp/mahlke92.ps.gz

Correction of a Memory Management Method for Lock-Free Data.. - Michael, Scott (1995) (Correct) (9 citations)
test-and-set operations together atomically. The **second** race condition arises from allowing a shared
common solution is to associate a modification **counter** with a pointer, to always access the **counter**
hypatia.dcs.qmw.ac.uk/data/edu/cs.rochester.edu/systems/95.tr599.Memory_management_for_lock-free_data_structures.ps.gz

Toward Simulated Evolution of Machine-Language Iteration - Huelsbergen (1996) (Correct) (7 citations)
This can directly increase GP's efficiency. **Second**, we are turning our attention toward harder
Instruction Comment 0: Cmp(R 2 ,R 3)compare **counter** R 2 to zero (R 3 =0) 1: Je(4) if **counter** zero,
For an integer register machine with an addition **instruction** as its sole arithmetic operator, we show that
netlib.bell-labs.com/who/lorenz/papers/gp96.ps

Optimizing ML with Run-Time Code Generation - Leone, Lee (1995) (Correct) (113 citations)
best known for his defeat of Hannibal in the **Second** Punic War. His primary strategy was to delay
when applied to a filter program and a program **counter**, the result is a function that is parameterized
corresponding C programs. For example, the BSD **packet** filter interpreter, which the BSD kernel uses for
foxnet.cs.cmu.edu/~petel/papers/staged/mleone-pldi96.ps

Modelling Instruction-Level Parallelism for Software.. - Ali-Reza Adl-Tabatabai (Correct)
contains AGU, FPA, and FPM resources, while the **second** cycle contains only an AGU resource. The resource
specifies a loop closing operation (decrement a **counter** and jump to the beginning of the loop if the
Parallelism Jan. 1993, Orlando, FL. Modelling **Instruction**-Level Parallelism for Software Pipelining
www.cs.cmu.edu/afs/cs/user/gyl/www/ifip93.ps

Comparing Software and Hardware Schemes For Reducing the.. - Hwu, Conte, Chang (1989) (Correct) (17 citations)
in approximately 70% of the branches. However, a **second** delay slot could be filled only approximately 25%
Another, less-expensive scheme uses an up/down **counter** for prediction. J. E. Smith reports an accuracy
a common technique to increase throughput of the **instruction** fetch, **instruction** decode, and **instruction**
ftp.crhc.uiuc.edu/pub/IMPACT/conference/isca-89-branch.ps

Service Scheduling And Cac For Qos Guarantee In Future Pcs - Qiu, Mark (Correct)
Its Polling Token Is Generated Every 1=fl I **Seconds**. Class K Ms Rtts Class 1 Ms Rtts Class K Ms
The leaky bucket is characterized by a token **counter** and an RTT **counter**, referred to as BS Token
ABR (available bit rate) classes. Also, the term **packet** rather than cell will be used to represent a unit
www.cwc.uwaterloo.ca/tech_reports/cwc04.ps.gz

An Upper Bound on Delay for the VirtualClock Service Discipline - Figueira (1995) (Correct) (38 citations)
10 **packets** of length 0.01TC in any interval of T **seconds**, where C is the rate of the outgoing link of all

max s ,1 N ,D max s ,re f w i
in terms of throughput, end-to-end delay, and **packet** loss rate. The lack of strict performance
counter.cs.umd.edu/~rich/courses/cmsc818G-s98/papers/virtualclock.ps

Performance Comparison Of Video Transport Over ATM.. - Hossain, Kang, Horst (Correct)
feedbacks and network loss handling mechanisms **second**, the performance study for our video service
two interconnects. The appropriate user-level **packet** sizes for the served video are also determined
performance due to its dual issue (superscalar) **instructions** per cycle (Cycles/**Instruction** (CPI) can be
berserk.vlsi.uiuc.edu/people/ashfaq/ieee.mm97.ps

A Practical Electronic Voting Protocol Using Threshold .. - Baraani-Dastjerdi.. (1994) (Correct)
and Iversen [4] using encryption techniques. The **second** approach, proposed by Chaum [6]Ohta [10]
are voters, candidates, an administrator, and a **counter**. The scheme uses threshold encryption to
ftp.cs.uow.edu.au/pub/papers/1994/tr-94-13.ps.Z

Accounting for the performance of Standard ML on the DEC Alpha - Necula, George (1994) (Correct)
(3 citations)
64 Mbyte of main memory, and 2 Mbyte of **second**ary cache. The DECchip TM 21064-AA is the
for using the built-in hardware performance **counters**. The **counters** provide detailed information of
processor state during execution such as: total **instructions**, multiple-issue, stalls, cache behavior, and
www.cs.cmu.edu/~necula/alpha.ps.gz

The Case For Reliable Concurrent Multicasting Using.. - Levine, Lavo.. (1996) (Correct) (38 citations)
of reliable multicast protocols proposed to date. **Second**, we introduce Lorax, which demonstrates the
is not guaranteed, and a node may wish to keep a **counter** of how many times its parent has been deleted
members of a multicast group, such that (a) every **packet** from each source is delivered to each receiver
www.cse.ucsc.edu/research/ccrg/publications/brian.mm96.ps.gz

A New Binary Logarithmic Arbitration Method for Ethernet - Molle (1994) (Correct) (9 citations)
by the original designers of Ethernet. The **second** class is the power users, who run data-intensive
performance under overload provide a strong **counter**example to that claim (at least when the number
in the presence of back-to-back minimum-length **packets** was reported to use up 20% of the CPU [30]Of
ftp.cs.toronto.edu/pub/reports/csrg/298/report.ps.Z

Revisiting the COUNTER Algorithms for List Update - Albers, Mitzenmacher (Correct)
is that, in a straightforward implementation, a **second** pass through the list is required after each
Revisiting the **COUNTER** Algorithms for List Update Susanne Albers
www.research.digital.com/SRC/personal/Michael_Mitzenmacher/NEWWORK/postscripts/counter.ps.gz

Performance Analysis of Adaptive Location Management for Mobile .. - Wang, Chen, Ho (Correct)
drastic reduction in communication throughput [9]**Second**, frequent location updates incur extensive
a long handoff delay that leads to significant **packet** drop and throughput reduction. We have developed
www.seas.smu.edu/~wchen/mobile/palm.ps.gz

Dynamic Global Packet Routing in Wireless Networks - Kahale, Wright (1997) (Correct) (14 citations)
the first of which is the transmitter ,the **second** being the receiver)Our algorithms might be
Dynamic Global **Packet** Routing in Wireless Networks Nabil Kahale Paul
www.research.att.com/~kahale/papers/infocom97.ps

A Replay Mechanism for Massively Parallel Computer RWC-1 - Nobuyuki Ichiyoshi (1994) (Correct)
basic replay scheme that uses software **instruction counter** for relatively low-overhead recording of timings
processor coupled with a network switch. A message **packet** is sent out into the network by a make **packet**
in flexible computational tasks. Multiple-**instruction**, multiple-data stream (MIMD) type parallel
jisp.cs.nyu.edu/RWC/rwcp/papers/1994/B-21_183.ps.gz

Indicators for the Assessment of Congestion in TCP over ATM-UBR - Rahin, Kara (1997) (Correct)
being simulated. This turned out to be 100.01 **seconds** for LAN and 24.92 **seconds** for WAN. Each
the network load. At the source side, a timeout **counter** is maintained for the last unacknowledged
commonly used metrics, cell loss ratio (CLR) and **packet** retransmission ratio (PRR) two new indicators,
agora.leeds.ac.uk/scs/doc/reports/1997/97_42.ps.Z

The Pendulum Instruction Set Architecture (PISA) - Carlin Vieri (Correct)
where copy is initially clear 1 **Second**, control flow operations must be paired with an
Logic Nor Gpr[x] General Register X Pc Program **Counter** Mem[x] Memory Location X Table 1: **Instruction**
The Pendulum **Instruction** Set Architecture (PISA) Carlin Vieri May 5,
www.ai.mit.edu/~cvieri/pisa.ps

First 20 documents Next 20

Try your query at: Google (CiteSeer) Google (Web) CSB DBLP